UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/651,425	08/30/2000	Christopher Songer	003048.P008	2566

23363 7590 09/10/2004

CHRISTIE, PARKER & HALE, LLP
PO BOX 7068
PASADENA, CA 91109-7068

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	

DATE MAILED: 09/10/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/651,425	SONGER ET AL.	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *28 May 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-44* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-44* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *20040528*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 5/28/2004.

As indicated in Applicant's response, claims 1, 22, 43, and 44 have been amended. Claims 1-44 are pending in the office action.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-9, 19-30, and 41-44 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tseng et al., USPN: 6,009,256 (hereinafter Tseng), in view of Schlansker et al., USPN 6,408,428 (hereinafter Schlansker), and Kolchinsky et al., USPN: 5,535,406 (hereinafter Kolchinsky); and further in view of Miller et al., USPN: 6,175,948 (hereinafter Miller) .

As per claim 1, Tseng discloses a method of creating run time executable code, the execution using a processing element array, comprising:

partitioning a processing element (PE) array into a plurality of hardware accelerators (e.g. col. 1, lines 45-59; FPGA *chip*, col. 10, lines 2-22; col. 12, lines 45-55; Fig. 30-33 – Note: array of chips on to be reconfigured into circuits helping the execution of the kernel code is equivalent to array of PEs being partitioned in plurality of hardware accelerators)

decomposing a program source code (e.g. col. 58, line 62 to col. 59, line 50; Fig. 4, 26, 28; *Verilog, VHDL, parsing process, compiler 210* - col. 15, line 62 to col. 16, 38 -

Note: language that is programmed from human readable format then parsed and compiled into machine readable constructs are source program code, e.g. Verilog and VHDL) into a plurality of kernel sections (e.g. *code component* -col. 13, lines 27-36; *HDL code description, component*-- col. 11, lines 47- 64; *RTL level ... signals* – col. 60, lines 53-62; Fig. 29 – Note: HDL code component analysis is equivalent to decomposing into kernel sections; steps *301, 302, 304, 310*, Fig. 4)

(i) mapping said plurality of kernel sections into a plurality of hardware dependent model entities representation language (e.g. *hardware model, gate level,* – col. 11, lines 47- 64; *RTL* - col. 11, line 50 to col. 12, line 6; *hardware execution models* - col. 13, lines 36-40); and

generating a mapping between said hardware accelerators and said hardware dependent model entities configured to support run time execution (e.g. steps *306, 307, 309* - Fig. 4 ; Fig. 6; col. 19, lines 14-21; col. 21, lines 36-43) of the plurality of kernel sections (e.g. Fig. 25-30).

But Tseng does not explicitly disclose that the mapping in (i) is mapping of kernel sections into a plurality of hardware dependent executable code for execution on a plurality of hardware accelerators and that said hardware dependent executable code is to support runtime execution of the kernel sections. However, Tseng discloses a programming language derived from the hardware dependent model entities, e.g. gate level, for support of execution or simulation/debug on said plurality of hardware accelerators (e.g. Fig. 25-30; col. 11, line 50 to col. 12, line 6 – Note: RTL level signal used in program to effect signaling of a FPGA and support emulation/simulation/debug of hardware circuit is equivalent to mapped kernel sections code for executing hardware

accelerators in FPGA) and partitioning via analysis from a software kernel so as to map during compilation of the kernel into execution by a hardware accelerator among the element array (e.g. col. 11, line 47 to col. 12, line 15). Mapping of design model specification language into corresponding executable code for implementing or emulating a specific hardware embodiment was a known concept in the art of software/hardware emulation/simulation and design at the time of the invention. Analogous to the concept of mapping software into hardware partition by Tseng, Schlansker, in a method to find an optimum design for match hardware parameters and performance requirements to a particular type of processors, i.e. to translate into processor instruction code the corresponding parameterization specifics representing a hardware dependent requirements analogous to the mapping of kernels to hardware RTL language (or software/hardware partitioning) by Tseng, discloses mapping of instruction code sets via intermediate code against processor/hardware parameters or kernel sections as claimed (e.g. *parametization, opsets, opgroups* - col. 9, line 24 to col. 10, line 32; *AIR* - col. 19, line 23 to col. 21, line 4; Fig. 12-15). It would have been obvious for one of ordinary skill in the art at the time the invention was made to enhance the intermediate gate level program code as taught by Tseng with the mapping of such code into executable instruction groups as suggested by Schlansker, in case Tseng does not already include one such mapping, because this would help create a corresponding set of executable that best suits the hardware specification of the accelerators without extraneous use of resources, averting thereby hardware architecture/code operation non-compliance as intended by Schlansker.

Nor does Tseng specify forming a matrix describing different combinations of said hardware accelerators and said hardware dependent executable code, and variants to support run time execution of the kernel sections. Tseng, however, discloses mapping of hardware models or logic component to reconfigurable boards or circuit (e.g. Fig. 6; col. 21, line 63 to col. 22, line 8; step *309* - Fig. 4), mapping hardware dependent executable code (*RTL* - col. 11, line 50 to col. 12, line 6) and optimizing functions in interconnecting processing elements using a matrix (e.g. col. 25, line 49 to col. 26, line 15); hence the concepts of mapping hardware dependent code as well as using combination or variants of hardware configuration or reconfigurable array element are thereby disclosed. Further, Schlansker teaches corresponding of instruction sets to a particular processor (Fig. 1-3, 14) and use of template and matrices to map instruction groups or templates against execution constraints/parameters or data flow conditions (Fig. 17-20 – Note: execution conditions implies machine specific execution, hardware dependency); thus has also disclosed support via mapping of variants or combination of code instruction against hardware real world specific requirement. Further, Kolchinsky, in a method for processing applications using programmable logic device, e.g. FPGA, in executing a multitasking kernel with logic design configuration and encoded hardware algorithm coupling analogous to the hardware/software modeling and simulation process by Tseng, discloses using a programmable logic matrix array for mapping execution operations to criteria of the hardware control and configuration (e.g. col. 5, lines 35-65) to perform a specific algorithm. In view of Tseng's suggesting of the use of connectivity matrix and mapping of software logic or models to processing elements or circuits, it would have been obvious for one of ordinary skill in the art at the time the invention was made to use

the matrix array as suggested by Schlansker and taught by Kolchinsky and apply it to the descriptively mapping of hardware modeled from software components (designs as claimed) to the reconfigurable target circuits (accelerators as claimed) as taught by Tseng to support execution of kernel sections. One of ordinary skill would be motivated to do so because this would provide more visibility to the mapping process and added flexibility, hence cost-efficiency, for modification or correction in general and specially when the target architecture increases in complexity (e.g. Kolchinsky: col. 1, line 61 to col. 2, line 4).

Nor does Tseng explicitly disclose that the code created for runtime execution is for a processing element array; or that the matrix is for support of runtime execution of kernel sections by the processing element array. The use of partitioning by Tseng via hardware accelerators is being viewed as purported to expedite the emulation process implemented via the software kernel. The use of programmable array such that each accelerator or processing element therein is itself a processor in order to even accelerate this emulation process as suggested by Tseng (e.g. *EAB, DSP* -- col. 51, lines 27-34) was a known concept in the software-assisted hardware simulation/emulation or modeling/design process. Schlansker also discloses mapping executable for the target processor (see above). Miller, in a system using compiler to assist hardware design using modeling and mapping analogous to Tseng or Kolchinsky, discloses processing elements in a FPGA to be a DSP, partitioning of processing elements based on specifications of a model and defining of execution tasks for target processing elements (e.g. col. 6, line 6-19; col. 9, line 66 to col. 10, line 27; col. 10, line 59 to col. 11, line 49; Fig. 5-7). Based on the partitioning and accelerator approach by Tseng (in combination

with Schlansker/Kolchinsky), it would have been obvious for one of ordinary skill in the art at the time the invention was made to provide when resources allow, the processing element array implemented with processors as suggested by Miller, so that the kernel code as mentioned by Tseng can be partitioned and executed by the very processing elements as taught by Miller, because the benefits would be allow testing/diagnostics or optimization of code being imparted to these processing elements enabling anticipation and information on the behavior of those partitioned target hardware elements, hence additional rewriting or recreation of code for the target elements would be averted when the real target become instantiated and available (e.g. col. 5, lines 20-46; col. 8, lines 22-42).

As per claims 2 and 3, the combination Tseng/Schlansker/Miller further discloses partitioning into digital signal processors (re claim 2) and into bins (re claim 3) (Tseng: *EAB, DSP* -- col. 51, lines 27-34 ; *cluster* – col. 22, line 46-57; Schlansker: instruction groups – Fig. 16; Miller: col. 6, line 6-19)

As per claim 4, Tseng further discloses mapping includes mapping into multiple hardware contexts (e.g. col. 22, line 12-45; Fig. 28 – Note: grouping by gate netlist/Register Translation, or chip cluster logic/common clock is equivalent to mapping for multiple hardware context).

As per claims 5 and 6, Tseng further discloses mapping (re claim 5) a first set of invariants (e.g. *FPGA component lib 361*, *logic function 359* – Fig. 6) and that said first set of variants are produced (re claim 6) based on resource usage (e.g. RTL *358*, Fig. 6; *combinational component 303*, Fig. 4 – Note: register allocation is equivalent to resource usage).

As per claim 7, Tseng further discloses mapping a second set of variants of said designs configured to support multiple hardware configurations of one of a plurality of bins (e.g. circuit *design components, gate netlist* – col. 22, lines 1-23).

As per claim 8, Tseng further discloses mapping is performed by a place and route (e.g. steps *352, 353, 354, 355* – Fig. 6).

As per claim 9, Tseng further discloses the decomposition step is performed manually (*user* -- e.g. col. 17, lines 13-17).

As per claim 19, Tseng further discloses that mapping includes creating context dependent configurations (e.g. col. 22, line 12-45; components *901, 903, 904*-Fig. 28 – Note: configuring by gate netlist/Register Translation language, or chip cluster logic/common clock context is equivalent to mapping for multiple context dependent configuration; code components are equivalent to code context configurations).

As per claims 20 and 21, Tseng does not explicitly teach that the matrix used in the mapping is sparely (re claim 20) or fully (re claim 21) populated; but discloses the connectivity matrix and mapping of FPGA circuits to hardware models (e.g. Fig. 7, 16; col. 25, line 49 to col. 26, line 15), the component type used for defining partial or generalized connectivity in the array of processing elements (e.g. col. 18,lines 20-41). Combining Tseng's teaching with Schlansker/Kolchinsky's use of a matrix to map design logic to configuration resources as mentioned in claim 1, the limitation on such matrix being populated as claimed herein would have been obvious because of the same rationale mentioned in claim 1; and also because Tseng's component type analysis as mentioned above would imply sparsely or fully populating of the matrix as mentioned in claim 1.

As per claim 22, this claim is a system claim corresponding to claim 1 above and includes most of the limitations therein using Tseng's disclosure, namely:

a processing element array or plurality of hardware accelerators partitioned (e.g. col. 1, lines 45-59; FPGA *chip*, col. 10, lines 2-22; col. 12, lines 45-55);

plurality of kernel sections (e.g. *HDL language, code component* -col. 13, lines 27-36; *HDL code description, component*-- col. 11, lines 47- 64)

created from a program source code (e.g. col. 58, line 62 to col. 59, line 50; Fig. 4, 26, 28; *Verilog, VHDL, parsing process, compiler 210* - col. 15, line 62 to col. 16, 38) for execution on said plurality of hardware accelerators (e.g. Fig. 25-30);

plurality of hardware dependent model representation language (e.g. *hardware model, RTL, gate level,* – col. 11, lines 47- 64; *hardware execution models* - col. 13, lines 36-40);

(1) mapping of accelerators and kernel designs for run time execution (e.g. steps *306, 307, 309* - Fig. 4 ; Fig. 6; col. 19, lines 14-21; col. 21, lines 36-43); hence is rejected using the corresponding rejection as applied in claim 1 above, using Tseng's teachings.

However, like in claim 1, Tseng does not explicitly teach the mapping in (1) is mapping of kernel sections into a plurality of hardware dependent executable code for execution on a plurality of hardware accelerators and that said hardware dependent executable code is to support runtime execution of the kernel sections. But this limitation has been addressed in claim 1 using Schlansker.

Nor does Tseng disclose a matrix used for said mapping in (1), but the matrix limitation including describing different combinations of accelerators, hardware dependent executable code, and variants has been addressed by Tseng combined with

Schlansker/Kolchinsky as mentioned in claim 1 above; and is herein rejected using the same rationale set forth therein.

Nor does Tseng explicitly disclose that the code created for runtime execution is for a processing element array; or that the matrix is for support of runtime execution of kernel sections by the processing element array. This limitation has been addressed in claim 1 from above using Tseng' suggestion, Schlansker's and Miller's teachings.

As per claims 23-30, these are system claims corresponding to claims 2-9, respectively; hence, are rejected using the corresponding rejections set forth therein, respectively.

As per claims 41-42, these claims are similar to claims 20-21 above, respectively; hence are rejected herein using the same grounds set forth therein.

As per claim 43, this claim is a computer-readable medium version of claim 1, above hence includes all the step limitations therein and is rejected herein using the same corresponding rejections set forth therein.

As per claim 44, this claim is a system version of claim 1, above hence includes all the step limitations therein and is rejected herein using the same corresponding rejections set forth therein

4. Claims 10-17, and 31-38 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tseng et al.,USPN: 6,009,256, in view of Schlansker et al., USPN 6,408,428, and Kolchinsky et al., USPN: 5,535,406, and Miller et al., USPN: 6,175,948, as applied to claims 1 and 22 above, and further in view of Trimberger, USPN: 5,752,035 (hereinafter Trimberger).

As per claim 10, Tseng does not specify that the decomposition step is performed by a software profiler; but discloses deriving of gate combinational elements based on type analysis (e.g. col. 18, lines 22-41; step 302, Fig. 4) for network array execution path differentiation; and profiling functions to optimize the cost for placement of processing elements in the array (e.g. col. 23, line 58 to col. 23, line 21; Fig. 6). Schlansker, in the method (re claim 1) to match an instruction set for a particular accelerators against data flow conditions and parametric requirements, discloses organizing instruction code into set and groups for evaluation against operational statistics (e.g. Fig. 16-20 – Note: collection of statistics and metrics for performance evaluation is equivalent to profiling) and Trimberger, in a method for compiling and executing programs using re-programmable logic accelerator sets analogous to the FPGA accelerators used in Tseng's method (combined with Schlansker/Kolchinsky) for simulation, discloses the use of profiling to detect the most frequently executed code and replace it with programmable instructions unit, or accelerator set (e.g. col. 6, lines 31-56). It would have been obvious for one of ordinary skill in the art at the time the invention was made to use a profiler as suggested by Schlansker and taught by Trimberger to detect optimizable portions of code for submission into accelerator logic units as taught by Trimberger and combine it with the analysis and optimization techniques above-mentioned by Tseng (combined with Schlansker/Kolchinsky) because this would provide additional performance improvement of the program execution/simulation performed in Tseng's invention (combined with Schlansker/Kolchinsky).

As per claim 11, Tseng discloses that the decomposing step includes executing code from program source code and monitoring timing of said execution (e.g. col. 20,

line 4 to col. 21, line 24; Fig. 6; *monitor($time...), test-bench component (monitor) 906* – Fig. 26,28; *Eval Timer 1004*, Fig. 23).

As per claims 12 and 13, Tseng discloses utilizing set of test data (e.g col. 13, lines 41-47; *evaluate test-bench components*, steps *333, 337*-Fig. 5; Figs. 29-30) in the execution of the simulation; but does not specify using test data (re claim 12) in the decomposing step nor does Tseng specify that (re claim 13) said monitoring includes determining functions that consume a significant portion of execution timing. But the limitation of using a profile analysis to determine code portions occupying significant execution time prior to optimization has been addressed in claim 10 above in view of Schlansker and Trimberger's teachings. In view of Tseng's (combined with Schlansker/Kolchinsky's) teaching to partition software simulation and hardware simulation (Figs. 29-30) using iterative test-bench components testing as mentioned above, it would have been obvious for one of ordinary skill in the art at the time the invention was made to modify the decomposition step disclosed by Tseng (combined with Schlansker/Kolchinsky) and combine the profiling technique (so suggested by Trimberger) and the use of test data (as taught by Tseng) for detecting the code portion susceptible for further optimizing, or for accelerating in Tseng's (combined with Schlansker/Kolchinsky) method , because this would enhance further the incremental the code optimization or simulation cycle as taught by the combination Tseng (combined with Schlansker/Kolchinsky's teaching).

As per claim 14, Tseng discloses a decomposition step with identifying kernel sections by identifying regular structures (*component type* – col. 17, lines 24-34).

As per claim 15, Tseng discloses identifying kernel sections by identifying sections with a limited number of inputs and outputs by way of basic blocks (e.g. col. 44, lines 55-65; Figs. 17,18) of the hardware model, which are register component types fetched from the component type analysis, i.e. decomposition step (e.g. col. 18, lines 20-41—Note: basic blocks are those whose inputs and outputs connection are in very limited number).

As per claim 16, Tseng further discloses identifying kernel sections by identifying sections with a limited number of branches (e.g. col. 18, lines 20-41; col. 44, lines 55-65 - Note: a skill in the art would view basic blocks as those having limited number of branching)

As per claim 17, Tseng discloses decomposing by identifying sections that are overhead sections (e.g. software model 215, Fig. 3; col. 16, lines 28-32).

As per claims 31-38, these claims are system claims corresponding to claims 10-18, respectively, hence, are rejected herein using the corresponding rejections set forth therein, respectively.

5. Claims 18, 39, and 40 are rejected under 35 U.S.C. 103(a) as being unpatentable over Tseng et al., USPN: 6,009,256, in view of Schlansker et al., USPN 6,408,428, and Kolchinsky et al., USPN: 5,535,406, Miller et al., USPN: 6,175,948; as applied to claims 1 and 22 above, and further in view of Mirsky et al., USPN: 6,457,116 (hereinafter Mirsky).

As per claim 18, Tseng (combined with Schlansker/Kolchinsky) does not disclose that mapping includes creating microcode. Mirsky, in the method for configuring array of processing elements in response to state data submitted analogous to

the simulation and testing of test-bench components applied to the circuit of processing elements disclosed by Tseng, discloses the use of microcode (e.g. col. 10, lines 21-32). In view of the use resource-constrained processing elements such as chips and DSP as suggested by Tseng (col. 51, lines 27-34) for partitioning, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement microcode to the processing element such as taught by Mirsky to Tseng's (combined with Schlansker/Kolchinsky) mapping of hardware configuration to the design circuit for carrying out the simulation process as taught by Tseng. One of ordinary skill in the art would be motivated to do so because this would alleviate memory storage in small device used as PEs in the array of Tseng's system (combined with Schlansker/Kolchinsky), and further improve resource usage and time-efficiency.

As per claim 39, this claim corresponds to claim 18 above, hence, is rejected herein using claim 18 rejection as set forth above.

As per claim 40, Tseng discloses code including context dependent configurations (e.g. *register component 901, clock component 903, Test-bench component 904* – Fig. 28 – Note: all component-related code with configuration for variables are equivalent to context dependency configurations); but Tseng fails to disclose microcode in place of code. Such limitation has been addressed in claim 18 above and is rejected herein with the same ground of rejection set forth therein.

Response to Arguments

6. Applicant's arguments filed 5/28/04 have been fully considered but they either moot in view of new grounds of rejection or not persuasive. Following are the most

representative points raised from the arguments and corresponding response by Examiner.

(A) Applicants have submitted that Tseng does not teach or suggest "matrix describing different combinations ... by the processing element array"(Appl. Rmrks, pg. 10, 3rd para); that Tseng's system does not generate code 'to support runtime execution ... element array ... Rather, the executable ... by the processor/workstation 240, not by the FPGA chips' (Appl. Rmrks, pg. 12, 2nd para); and that there is no 'plurality of hardware accelerators partitioned from the processing element array" (Appl. Rmrks, pg. 12, 3rd para).

The rejection has pointed out what Tseng discloses and what Tseng explicitly lacks so that the missing features are to be provided from another reference. As for Tseng's not disclosing partition of a PE array, it is noted that the FPGA is global proposed hardware representation of the target system to be designed and simulated by Tseng whereas the chips or combinations thereof are the elements. Tseng's system has created a software kernel via whose analysis the partitioning into some specific accelerators is effected in order to carry out the emulation process code provided from that kernel; and the FPGA chips being modeled are tested from this kernel using the accelerators (or combination of chips) being partitioned to help accelerate the test or bench-mark diagnostics of the emulation process (see Fig. 4, 5 and related text). The hardware partitions are set forth during analysis of the compiled kernel (see rejection) and belong to the target hardware or model representation (see Fig 30-33) as they contribute in carrying out the execution instructions of the kernel. Hence they belong to a partition process made in the physical context of the FPGA used as target. In other

words, the accelerator partitions being implemented are part of the target process element array or hardware model, each partition contributing its share of kernel enabling tasks. Hence Tseng discloses the partitioning of the FPGA because the above teaching reads on partitioning the PE array into a plurality of hardware accelerators.

As for the argument that Tseng does not generate code to be executed by the processing element array, the rejection as put forth does point out how the partition with accelerators suggests providing expediency in executing Tseng's emulation or testing processes or tasks derived from the software kernel. The motivation as to provide processing element so that such element would be able to execute those instructions such as taught by Schlansker or Miller are also provided. Tseng's teaching thus is only suggesting the use of part of the PE array (see Fig. 30-33) to carry out the testing process; and since the rejection is a 103(a), such suggestion is then supported by additional teachings from one or more other references in order to establish a prima facie type rejection along with a motivation to combine as viewed by a one of ordinary skill in the art. Thus, Tseng's teaching is to be taken not separately but in conjunction with what has been suggested in Tseng's and what is taught in the other references in accordance with the rationale set forth by the rejection. Applicants have to show how the combining as set forth in the rejection would be inappropriate or would not arrive at a reasonably successful result; and cannot contend with depicting what any one reference taken individually is lacking or not suggesting.

(B) Applicants have submitted that Schlansker or Kolchinsky does not teach or suggest the limitations of claim 1, 22, 43, 44 (Appl. Rmrks, pg. 12, bottom, pg. 13, top). As amended, the claims necessitate new grounds of rejections and the arguments should

become moot. Again, Applicants have taken each of the references separately in order to point what is missing therein but most importantly have not established why the combination as put forth in the rejection would be improper and why so for very specific reasons. In other words, in response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.,* 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

Information Disclosure Statement

7. The information disclosure statement filed 5/28/2004 fails to comply with 37 CFR 1.98(a)(2), which requires a legible copy of each U.S. and foreign patent; each publication or that portion which caused it to be listed; and all other information or that portion which caused it to be listed. It has been placed in the application file, but the information referred to therein has not been considered.

The non-patent documents (6 of them) listed in the form PTO-1449, pg. 1-2 are not found to come with a legible copy for each. The documents are hence not considered, i.e. those not being initialized by Examiner or marked with 'NC'. Applicants are urged to resubmit these copies if such non-patent documents are deemed of some import for consideration.

Conclusion

8. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP

§ 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 872-9306 (for formal communications intended for entry)

or: (703) 746-8734 (for informal or draft communications, please consult Examiner before using this number)

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

VAT
August 26, 2004

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100